The Small World Web of AI

Noa Zilberman
University of Oxford
Oxford, UK

Alexander Jackson
University of Oxford
Oxford, UK

Abstract

The rise of generative AI has transformed many fields, including networking. While generative AI is already used for network management and operation, little was done to fundamentally change the way we use the web. In this paper, we make the case for reimagining the web in the age of AI. With just minor changes to HTTP, we demonstrate that web content can be distributed as prompts turned into content on end user devices. This new small world web (SWW) of AI reduces storage demands and network load, and has the potential to improve Internet sustainability over time.

CCS Concepts

• **Information systems** → **World Wide Web**; • **Networks** → **Network protocols**; • **Computing methodologies** → *Artificial intelligence*; • **Hardware** → *Power and energy*.

Keywords

WWW, Generative AI, HTTP, Sustainability

ACM Reference Format:
Noa Zilberman and Alexander Jackson. 2025. The Small World Web of AI. In *The 24th ACM Workshop on Hot Topics in Networks (HotNets '25), November 17–18, 2025, College Park, MD, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3772356.3772420

1 Introduction

Have you ever experienced that sense of déjà vu online? It's the feeling you get when every food delivery menu looks exactly the same, and every travel blog seems to describe the same hiking trail.

A lot of the web content that we see is generic. Either purposefully, using stock images, or unintentionally, using boilerplate text to create website pages. While some of this content is useless, other is useful but plain. Plain enough that even a simple generative AI (GenAI) model can create it. So possibly we should let AI create it?

HotNets '25, College Park, MD, USA

ACM ISBN 979-8-4007-2280-6/2025/11
https://doi.org/10.1145/3772356.3772420

In this paper, we propose that by using GenAI to generate content from prompts on end-user devices, we can save storage space and data transmission, and potentially improve sustainability. Our focus is not on the AI and content generation, but rather on the underlying networking protocols and the changes required to support a shift to edge-generated web content.

Researchers are already exploring GenAI for networking (e.g., [27, 31, 35, 37, 59]). Additional work focuses on content generation, including for webpages (e.g., [4, 26, 30, 54, 55, 60]). However, to truly benefit GenAI, we need to redesign aspects of the Internet.

In this paper, we make the following contributions:

- We introduce a concept for a world wide web enabled by GenAI and explore the potential benefits.
- We propose a modification to HTTP to support GenAI-enabled web pages, and introduce a prototype implementation of client and server side.
- We evaluate our prototype's image and text generation, measuring its compression ratio, energy efficiency, and content similarity. Our results reveal current benefits and potential future gains.

GenAI is still in its infancy but rapidly developing. While technology is not ready yet for full deployment, this work proposes a pathway forward and predicts a future where user-tailored content is generated at the edge, leading to a scalable and sustainable small world web of AI.

2 SWW: Reimagining WWW

In a world where GenAI is already used to turn events into summarized bullet lists, and prompts into generated images, videos and sound, there is a need to adjust the web to the changes in content and its use. Our insight is that GenAI can be used to *reduce* the data transmitted over the network, storing and sending less bits from servers to users.

2.1 Basic Usage

As an example of a simple use case, assume visiting a travel blog webpage. The page will include some generic text about traveling and a few stock images of landscapes. It may also include some unique content, such as the details of a specific hiking route or pictures taken during the hike. Currently, the entire text will be sent from the server to the user, as well as image files (e.g. jpeg files).

In a re-imagined web page, the server stores a baseline webpage with prompts that should be used to generate content. Only unique content, such as pictures from the specific hike, are stored on the server and all other content is turned into prompts. Route-specific text is either kept as is, or turned into bullet points that can be used in a prompt to generate the relevant text without loss of information. When a user requests the webpage, the server sends the baseline webpage, along with the prompts. On the user side, the browser parses the incoming webpage, and uses the prompts to generate the content on the user's device. Unique content files are fetched, same as today, and presented to the user.

The example above demonstrates three potential benefits:

- Reduced network load - with the majority of content transferred being text-based rather than media, the amount of data transferred over the network is reduced, and the overall load on the network infrastructure is lower.
- Reduced storage requirements - web servers will need to store only the prompts required to generate content rather than the content itself.
- Improved sustainability - with less stored and transferred content, there are potential energy and carbon savings to be made.

We explore and evaluate the potential benefits in §6.

2.2 Content Distribution Networks

We identify Content Distribution Networks (CDNs) as a place where SWW is likely to have a large impact. The reason is the replication of content across sites that leads to increased storage demands. By moving to storing prompts rather than storing content, CDNs can reduce storage requirements.

Currently, one of the limitations for an immediate adoption of SWW is the ability of end user devices to generate content from prompts quickly and with high quality (§6). CDNs allow an intermediate solution: media is sent from the content provider to caching locations or edge servers as prompts, and only the prompts are saved at the edge. At a request of a user, the edge server uses the prompt to generate the content and sends it to the requester. This approach maintains the storage benefits, but loses data transmission benefits. There's also a potential energy and carbon emissions trade off when running at the edge or on end-user devices, which we consider in §6.

While so far we discussed content *generation*, another option is content *upscaling*, such as turning small images into large, high resolution ones. By using content upscaling, the storage requirements of unique content can be reduced as well. Content upscaling is also usually faster than content generation, with sub-second inference [58].

2.3 Personalized Content

Generating content on end-user devices also means that there is an opportunity to generate *personalized* content on these devices. The generation algorithm can use as an input information about users' background, preferences and hobbies and create content that is likely to increase the user's engagement with the website or the product.

This personalized approach is likely to very attractive, however it has a potential for harm, not only from malicious actors but also by creating an echo chamber and amplifying other online harms [19, 20, 22]. We therefore highlight this as a major concern as an element that needs to be addressed prior to deployment. We urge the wider web community to consider the harms of personalized content in SWW.

2.4 Supporting SWW

Enabling support for SWW requires a relatively small set of changes. First, the communication protocol needs to support the functionality. In §3 we demonstrate this can be done with a minor modification to HTTP. Second, webpages need to be modified to indicate and support content generation. We suggest some ideas for implementation in §4. Last, the webserver and client (e.g., web browser) need to be modified, which is discussed in §5.

Our prototype implementation[1], which supports all of the above, required less than a thousand lines of code.

3 Modifying HTTP

To demonstrate SWW, we modify HTTP/2 [11, 52] to support advertising of client-server capabilities. To this end, we leverage the `SETTINGS` frame that allows peers to exchange configuration parameters during connection setup.

The SETTINGS frame is defined in RFC9113 [52]. It affects the entire connection, not just the stream it is sent across. Each entity stores the latest settings it receives from its peer and uses them to structure appropriate messages across all streams. The Settings structure consists of a 16-bit setting identifier and a 32-bit value for the setting. HTTP/2 defines six reserved SETTINGS parameters, such as MAX_FRAME _SIZE and MAX_CONCURRENT_STREAMS, with developers able to add additional settings. A SETTINGS frame is acknowledged by an empty settings frame with an ACK flag, otherwise the connection is void and treated as PROTOCOL_ERROR. A recipient receiving an unrecognized setting ignores it, meaning that recipients without GenAI capabilities can continue to operate as before.

In our prototype, we add a new setting value, `SETTINGS_GEN_ABILITY (0x07)`. This setting informs the recipient of

[1]Available at https://github.com/ox-computing/SWW-AI

a sender's ability to implement client-side content generation. The identifier is 0x07 (as the first unreserved value, for prototyping purposes) and the value is set to 1.

In any case other than both server and client having SETTINGS_GEN_ABILITY set to 1, default (unsupported) behavior will be assumed. In an exchange between a participating entity (supports extension) and non-participating entity (does not support extension), the participating entity will fall back to default as it realizes the other does not support the extension. The non-participating entity will remain naïve and continue to communicate over normal HTTP/2.

While currently a binary value is used to indicate support or lack of support, the 32-bit field can be used negotiate more complex support options, such as upscale-only.

3.1 HTTP/3

The initial choice to use HTTP/2 rather than HTTP/3 [13] was due to the relative ease of modifying and deploying it and its upper layers. Moreover, our initial design assumed that more complex changes would be required in the protocol and client/server, which would have been harder to support in HTTP/3. Since HTTP/2 still accounts for about 50%-60% of web traffic [12, 15], this was a reasonable assumption.

Still, as HTTP/3 adoption is increasing, future SWW will require HTTP/3 support. We believe that similar use of SETTINGS under HTTP/3 can allow to advertise client-server GenAI capabilities.

3.2 Video Streaming

Video streaming protocols, such as HTTP Live Streaming (HLS) and MPEG-DASH, run on top of HTTP. The proposed modifications to HTTP for web pages can be applied also to negotiate generation abilities also for video streaming.

Experimenting with the generation of video on end user devices is not in the scope of this work. Beyond technological considerations, legal constraints should also be considered (e.g., agreements with professional guilds [3]).

Quick and semantically correct generation of complete, long videos on user devices is still some time away. However, frame rate boosting, e.g., from 30fps to 60fps, is a likely early use case. Client-side video upscaling, including frame rate boosting and resolution improvement, is already available using GPU features like NVIDIA's RTX Video Super Resolution [42] or AMD's Fluid Motion Frames [7]. However, client-side upscaling is currently not a feature visible to the content provider.

In SWW, client devices can negotiate with the video server generation abilities before content is sent, similar to web pages. Sending content at a lower frame rate or lower resolution has a direct effect on data savings: moving from 60fps to 30fps will half the data, and from 4K to high definition can save 2.3× data, turning 7GB/hour into 3GB/hour [41]. The evaluation of this approach is left for future work.

4 Webpage Design

We consider several aspects of webpage design for SWW. First, how should the webpage itself be changed, demonstrated through a change to html. Second, how should webpages be generated at scale, and third, mechanisms for easing adoption and deployment.

4.1 HTML Parser

To support generated content, we add in our prototype a class called *generated content* which has two fields: content-type and metadata. Content-type identifies the type of generated content, currently supporting either "img" or "txt". Metadata is a json dictionary used to store metadata needed to generate the content. Examples of metadata fields include the prompt or width and height for images. These metadata fields vary between different types of content.

The HTML Parser extracts the metadata and passes the information to a media generator object, alongside a preloaded image generation pipeline, in order to generate the actual content. Once content is generated, the divisions in the HTML are replaced with accurate paths to images, or the actual body of text for text expansion tasks. The choice to preload the image generation pipeline from a library (for example, a Diffusers library) is for performance optimization. Since it is a large object, it would otherwise need to be repeatedly deleted and reloaded within the media generator every time it is invoked.

The media generator has two roles: parsing the passed metadata and invoking content generation using the parsed information. The media generator has two generation subroutines, one to generate text and the other to generate images. In our prototype, the text-to-text models are accessed by sending requests to the Ollama API [16] using the requests library. The text-to-image models utilize Hugging Face's Diffusers Library [53].

An example of an HTML page before and after content generation is shown in Figure 1. Before (top) the HTML page contains the prompt required to generate a cartoon goldfish image, and after (bottom), it contains the pointer to the generated jpeg file.

```
<div class="generated-content"
     content-type="image"
     metadata='{"Prompt": "A cartoon goldfish", "Width": 441, "Height": 612, "Name": "Goldfish"}'>
    <p>Goldfish Image Placeholder</p>
    <div class="image-placeholder"></div>
</div>
```

```
<div class="generated-content"
    content-type="image"
    metadata='{"Prompt": "A cartoon goldfish", "Width": 441, "Height": 612, "Name": "Goldfish"}'>
    <img alt="Generated: A cartoon goldfish" src="../generated_images/Goldfish.jpg"/>
</div>
```

Figure 1: Top: HTML div before processing. Bottom: HTML div after processing.

4.2 Webpage Creation and Conversion

One bottleneck to rapid adoption is the number of webpages that already exist and require conversion in order to support SWW. The answer to this challenge is, unsurprisingly, AI. More specifically, using GenAI to turn image and text into prompts. A simple script that goes over a webpage can identify content, call a media converter to turn the object into a prompt, and replace the existing object with a *generated content* object.

While the approach above is useful and efficient, it has two limitations. The first is the quality of the conversion and the second is identifying which content should be converted and which is unique and should remain untouched. A promising step toward the first is prompt inversion, which generates prompts from images with the goal of maintaining high fidelity in the re-generated images [39]. We believe that the quality of conversion will improve over time, and in the short-term human intervention may be required to audit conversion results – a webpage editor.

An easy way to identify content that can be generated is by adding a dedicated feature to content management systems (CMS) and webpage builders. The feature would tag every content item as generatable or unique. This one-bit flag will be associated with every linked file. Text blocks can be similarly tagged. Webpage templates can have different default values for conversion tags.

On websites where content is often updated, moving from unique content to SWW will be rapid with new items being tagged for generation or not. Such sites, however, are likely to contain a lot of unique content (e.g., news items). Websites that contain more static content, such as companies' websites or blogs, are expected to gradually move to SWW, likely when they upgrade their content management system, which is often a tedious process that requires reviewing all webpages, regardless of the proposed feature. It is expected that some webpages will not be updated, and will retain their original, unmodified, content. Such pages, however, are less likely to be cached or frequently accessed.

5 Client and Server Design

5.0.1 Browser Integration. Support of SWW will require browser support. Our development started by examining support through a Chrome extension. However, this attempt was dropped for several reasons, the primary one being lack of hardware access. Extensions run in a sandboxed environment restricting GPU usage and memory, which are both essential for generative models [25]. Another reason was lack of libraries support, as extensions rely on JavaScript/WebAssembly, which are incompatible with Python-based libraries like PyTorch and Diffusers. A third reason was security constraints in extensions that restrict system calls, file access, and network operations required for inference. Last, even with WebAssembly, lack of CUDA/DirectML access would significantly slow down inference.

As a middle ground, integrating the system into a custom Chromium [46] build was also considered. This would provide greater resource access and deeper integration with generative models. However, this approach also has major drawbacks for prototyping, such as maintenance complexity, high resource demands with computationally intensive builds and extensive C++ dependencies, and security risks requiring ongoing audits and patching.

Due to these challenges, both approaches were not used for this prototype. Instead, a stand-alone application was used to fetch and render content. Nonetheless, Chromium integration remains a compelling option for future development toward a fully generative browser.

5.1 Generative Server

A simple generative server was designed using the Python3 asyncio library [21] to handle asynchronous requests from clients. This server is relatively simple and uses the H2 Library to communicate over traditional HTTP/2 with a client, including handling SSL context, whilst allowing flexibility over the individual settings frame. When clients connect, the server negotiates the generative ability using the modified HTTP/2 as discussed in §3. If the client's generative ability is confirmed, the server can serve the content in its generative form as indicated by the client. If the ability is not confirmed it will serve traditional content with no client-side generation expected. A server can choose to serve traditional content even if the client supports generative ability, for example to provide higher performance or based on the availability of renewable energy.

5.2 Generative Client

A generative client is slightly more complex than the server. The prototype's client relies on three main entities: a custom HTML parser to process received pages, a PyQT Graphical User Interface (GUI) for rendering sites, and the H2 connection library for managing connections to the server and the modified protocol. The client provides website connection functionality, rendering it for the user to interact with.

Typically, the generative client begins by establishing a connection to the server, followed by exchanging settings, advertising its generation ability and logging the server's ability. After this, the client can send a webpage request. As the client receives the HTML file, it parses it and generates content. Once parsing and generation are complete, the site is rendered in the GUI.

6 Early Days Results

We evaluate our prototype, trying to answer a few questions regarding the benefits of moving to SWW today and in the future. In particular, we address the following questions:

- What is the performance of the prototype?
- What is the quality of generated content?
- What are the storage and energy costs?

6.1 Evaluation setup

Our evaluation uses two machines: a laptop, representing the client/end-user, and a workstation, representing an edge web-server or a high-end client. The laptop is MacBook Pro, with M1 Pro CPU, 16GB LPDDR5 and 16-core integrated Apple GPU, FP16 precision, with no large text encoder/tokenizer and requires attention splitting. The workstation uses AMD Ryzen Threadripper Pro 5 CPU, with 128GB DDR5, two NVIDIA ADA 4000 GPUs, FP16 precision, with a large text encoder/tokenizer and no need for attention splitting.

6.2 Basic functionality and Performance

Basic functionality testing covered scenarios where both client and server support generated content, only one side supports generated content, and no side supports it. Except for the first scenario, in all other cases the communication defaulted to standard HTTP/2.

When the client does not support generative content, the server uses the prompt to generate the content before sending it to the client. This saves storage space, and avoids saving two copies of content (prompts and original files).

A qualitative example of a generated webpage is provided in Figure 2[2], with the original on the left and the generated on the right. The original page is the result of a search through Wikimedia Commons for "Landscape", which triggers 1.4MB of data for 49 images to be sent across the web.

The conversion process deployed an offline image-to-text model based on GPT-4V (via OpenAI) to find detailed prompts ranging from 120 characters to 262 characters. The constructed metadata was then sent over the modified HTTP/2 protocol and generated at the end host. As shown, the semantic meaning of each picture is conserved over this process, though the images are not identical.

Generating this page on the laptop took close to 310 seconds, or 6.32 seconds per image. On the workstation, this took around 49 seconds, or roughly 1 second per image, which is still considerable. However, new models aimed at speed turn generation into a real time experience [32].

A significant gain of this experiment was in data reduction: instead of sending 1400kB of data as images, only 8.92kB were needed, providing a compression factor of 157x. In a worst case scenario, using maximum metadata size of 428B[3], the compression ratio is still 68x.

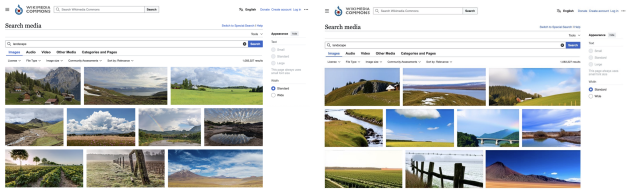

Figure 2: Left: Original Wikimedia search results for "Landscape", Right: Locally Generated Wikimedia search results for "Landscape".

An experiment of a similar nature explored text generation, by sending a newspaper article. This experiment, which at first resulted in a significant deviation in article length (discussed next), has taken 41.9 seconds on the laptop, more than ten seconds on the workstation, and provided 3.1× compression, from 2400B to 778B.

6.3 Quality of Generated Media

Next, a quantitative analysis of text-to-image and text-to-text was conducted. The analysis is provided here for indicative purposes only: as time goes by better models are developed, new fidelity metrics are introduced [17, 23, 24], and higher quality results can be achieved. The choice to show these results is in part to demonstrate the trade off in generation quality and generation time.

6.3.1 Text to Image. Our prototype uses Stable Diffusion 3 Medium, providing a balance between computational requirements and image quality. To evaluate the quality of generated images, we use two common metrics, ELO score and CLIP score[4]. ELO score [18] reflects the general opinion of a user base on the ability of a model to generate high quality images that adhere to the prompt. CLIP score uses OpenAI's CLIP model to measure similarity of an image to its text prompt [29]. We use CLIP scores as a quantitative measure, and ELO scores using data from Artificial Analysis [8] as a more qualitative metric.

Our evaluation compares a Stable Diffusion (SD) 3 Medium with SD 2.1 Base, SD 3.5 Medium, server-run DALLE-3. The CLIP score is evaluated using small images of 224×224 pixels with 15 inference steps.

The results, shown in Table 1, indicate that in terms of user opinion (ELO score), DALLE 3, SD 3 and SD 3.5 have relatively similar scores, with SD 2.1 performing significantly

[2]Original search results photos by Simon Koopmann, Martin Falbisoner, Pudelek, Petr Brož , Tomascastelazo, Basile Morin, Basile Morin, Dietmar Rabich, Dietmar Rabich and Diego Delso. Used under CC BY-SA license.

[3]400B to the prompt, 20B to the Name, and 4B to each height and width.

[4]We note the absence of universally accepted quantitative metrics that correlate well with human perception of image quality.

Model	ELO	CLIP	Laptop time/step	Workstation time/step
SD 2.1	688	0.19	0.18s	0.02s
SD 3 Med.	895	0.27	0.38s	0.05s
SD 3.5 Med.	927	0.27	0.59s	0.06s
DALLE 3	923	0.32	–	–

Table 1: ELO & CLIP scores, with time per step on a laptop and a workstation using 15 inference steps.

worse. For reference, the best performing model on the Artificial Analysis leaderboard at the time of writing was GPT-4o with ELO score of 1166.

The CLIP scores of SD 3 and SD 3.5 are almost identical, also when comparing laptop and workstation-based results, and are about 16% worse than DALLE 3, with SD 2.1 about 40% worse. As a baseline, the CLIP score of a randomly generated image (no prompt) was 0.09. While SD 2.1 performs poorly in terms of quality of generated content, it is significantly faster than other models. Generation time also sets apart SD 3 from SD 3.5, as it is 35% faster on a laptop and 13% faster on the workstation.

These trends remain as we scale inference steps from 10 to 60, with only minor changes to CLIP score and with generation time increasing linearly with the number of steps. As image size is increased, generation time is increased on the workstation relative to the number of pixels, but on the laptop it grows significantly beyond that for images of 1024×1024, reaching 310 seconds.

6.3.2 Text to Text. Four text-to-text models were evaluated: Llama 3.2, and DeepSeek-R1 1.5B, 8B and 14B. Three metrics are used to measure the quality of a model's text expansion responses. First, Sentence BERT (SBERT) Score [48] provides semantically meaningful sentence length embeddings, used here to compare bullet points semantic similarity to the paragraph of text. Second, Word Length Overshoot represents the percentage of words above or below the requested number of words. Finally, we measured content generation time.

All the models achieve SBERT mean scores ranging from 0.82 to 0.91, varying also with number of words. The overshoot in length reaches 20%, and while the mean of some models is close to 1.3%, the 25th and 75th percentile are in most cases over 10%. DeepSeek R1 8B, which is our model of choice, has a consistently high SBERT score and small length deviation across the data compared to smaller models like DeepSeek R1 1.5B.

Generation time ranges from 6.98s to 14.33s on the workstation, and from 16.06s to 34.04s on the laptop, but has only a weak dependence on the length of the generated text, e.g., 50 words text takes longer than 100 and 150 words text for three of the models. The performance benefit of running on a workstation is only 2.5×.

6.4 Compression and Energy

Table 2 shows the storage savings for different types of media, by savings metadata rather than raw files, the generation time of each element, using SD 3 Med and DeepSeek-R1 8B, and the energy required for generation. As the table shows, the bigger the image, the higher image compression ratio. However, generation time might be long. To put things in proportion, sending a large image on a typical 100Mbps link would take about ten milliseconds, while image generation on the workstation would take 620× longer. We assert that moving to faster models, aimed at reducing generation time [32, 33, 56], is required to support SWW.

At this point, one can compare the energy required to transmit content to the data required to generate the content. However, currently the network energy consumption is dominated by static power that is not affected by network activity. Therefore, in the near future, moving to SWW may have limited effect on network energy consumption. To understand the scope of network transmission compared to media generation, we use Telefonica's 2024 energy consumption per unit of traffic, which was 38MWh/Petabyte, or 0.038Wh/MB, meaning that a large image would cost roughly 0.005Wh to transmit, 2.5% of current workstation generation.

A different benefit of compression is the reduction of embodied carbon. Storage devices have a high environmental toll [50], amounting to $6 - 7kg/CO_2e$ per terabyte of SSD [34, 38]. With exabyte scale storage [45], even modest compression can save millions of kg/CO_2e.

7 The Long Road Ahead

Is It Worth It? Our evaluation results are not encouraging: currently, generating content at the edge takes too long and does not save energy. Still, we are optimistic that this will change by the time standards and commercial products support SWW. First, as AI models are evolving quickly, and already some models perform better (CLIP, ELO) and generate faster than SD 3.5 Medium [33]. Second, the recent year has seen a boost in the development of accelerators for inference tasks, as those are likely to improve consumer products too. Last, content providers have strong incentives to move to SWW model: significant storage space is saved, and with generation done on client-side energy costs are lowered.

While the ratio of web traffic out of overall Internet traffic is dropping, it is still significant. Web browsing from mobile devices alone amounts for 2-3 Exabytes/month [51, 57]. Reducing this number by approximately two orders of magnitude, as indicated in §6, will lower this number to tens of Petabytes/month.

Next Steps The work presented in this paper is preliminary. Adding support for more protocols, such as HTTP/3, is needed, as well as IETF standardization. Media generation

Media	Size[B]	Metadata [B]	Compress. Ratio	Laptop Gen.[s]	Laptop Energy[Wh]	Workstation Gen.[s]	Workstation Energy[Wh]
Small Image (256×256)	8,192	428	19.14	7	0.02	1.0	0.04
Medium Image (512×512)	32,768	428	76.56	19	0.05	1.7	0.06
Large Image (1024×1024)	131,072	428	306.24	310	0.90	6.2	0.21
Text Block (250 words)	1,250	649	1.93	32	0.01	13.0	0.51

Table 2: Generation time and energy consumption for typical small, medium and large images and 250 words text.

models need to be continuously updated. Moreover, MCP [9] was announced after work on the prototype has started, and was not integrated yet. Adding browser support is a next step. Model updates will likely be distributed as part of browser updates. Negotiating models is another aspect to consider.

Generation on Mobile Devices To achieve maximum impact, SWW requires generation on mobile devices. These devices are resource constrained, aimed at low power consumption, and often missing the required hardware acceleration capabilities. However, a change is coming. Companies such as Samsung [49], Apple [43], and Qualcomm [47] are introducing acceleration solutions for on-device generative AI. In parallel, significant AI research is focused on developing smaller, lighter and more efficient models for on-device generation [5, 36, 61].

New Opportunities Some new opportunities arise from SWW, ranging from browser design to efficient client-side accelerators. One interesting aspect is that of stock photos, as these will mostly become prompts. Possibly in a few years' time we will see stock prompts companies emerge. The conversion of vast amounts of existing web content to prompts is another challenge, especially when information fidelity needs to be preserved. In particular, identifying existing content that should not be AI generated is a challenge.

Ethics and Trust As mentioned in §2.3, there are significant ethical concerns around the use of SWW and the generation of personalized content. These need to be thoroughly studied. However, even without SWW, personalized content may be generated on the sender side, raising similar concerns. Another question relates to copyrights, as a lot of content will be reduced to prompts and then generated. Possibly content sharing licenses will be updated to allow use on SWW.

The trustworthiness of generated data is another aspect that needs to be carefully studied. This is not only a problem of the generated content diverging semantically from the original, but also of verifying generated content on end-user devices. Such verification should be accompanied by other mechanisms for trustworthy AI [10, 14].

Sustainability One of the original motivations for this work was sustainability, as reducing data storage and transmission can lead to reduced carbon emissions. While our results suggest that this currently is not the case for operational emissions, there are other exciting opportunities. First, as reducing storage space saves embodied carbon emissions. Second, as traffic reduction on the network provides more flexibility in cache placement, without breaching backbone traffic constraints. While the main limitation to cache location was often the latency to the user, in SWW the network latency is a minor problem compared with other major challenges.

8 Related Work

Content generation is currently widely explored, including in the context of generating web content, e.g., [30, 54, 55, 60]. AlDahoul et al. [4] explored generation of webpages images using stable diffusion using existing alt-text in html pages. Hassan et al. [26] explored compression ratio of images under different generation conditions. Several works considered the use of GenAI to increase webpage accessibility (e.g., [1, 28, 44]). Dinzinger et al. [19] surveyed ways to increase data sovereignty on webpages related to use by GenAI. To the best of our knowledge, no prior work addressed the network-protocols aspect of webpages using GenAI.

The concept "Small World Web" was previously noted in the context of small world networks, when studying the world wide web [2, 6, 40].

9 Conclusion

This paper introduced SWW, using GenAI to recreate webpages media on end-user devices, reducing storage requirements and data transmission demands. The paper suggested that with minor changes HTTP/2 can support SWW, and shown an integration with HTML. Preliminary results indicate that significant storage savings can be achieved, but currently with high energy costs and long generation time. The emergence of new low-power accelerator technologies will make SWW a sustainable, efficient solution.

Acknowledgments

This work was partly funded by NSF CBET-UKRI EPSRC TECAN (EP/X040828/1). We thank Tsahi Zilberman for the original idea, and Eve Schooler for her constructive feedback.

References

[1] Patricia Acosta-Vargas, Belén Salvador-Acosta, Sylvia Novillo-Villegas, Demetrios Sarantis, and Luis Salvador-Ullauri. 2024. Generative Artificial Intelligence and Web Accessibility: Towards an Inclusive and Sustainable Future. *Emerging Science Journal* 8, 4 (2024), 1602–1621.

[2] Lada A Adamic. 1999. The small world web. In *International Conference on Theory and Practice of Digital Libraries*. Springer, 443–452.

[3] SAG AFTRA. 2025. Artificial Intelligence, Contracts and Industry Resources. https://www.sagaftra.org/contracts-industry-resources/member-resources/artificial-intelligence [Accessed 14-October-2025].

[4] Nouar AlDahoul, Joseph Hong, Matteo Varvello, and Yasir Zaki. 2025. Towards a World Wide Web powered by generative AI. *Scientific reports* 15, 1 (2025), 7251.

[5] Keivan Alizadeh, Seyed Iman Mirzadeh, Dmitry Belenko, S Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. LLM in a flash: Efficient large language model inference with limited memory. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 12562–12584.

[6] Luıs A Nunes Amaral, Antonio Scala, Marc Barthelemy, and H Eugene Stanley. 2000. Classes of small-world networks. *Proceedings of the national academy of sciences* 97, 21 (2000), 11149–11152.

[7] AMD. 2024. AMD Fluid Motion Frames. https://www.amd.com/en/products/software/adrenalin/afmf.html [Accessed 14-October-2025].

[8] Artificial Analysis. 2025. Text-to-Image Model Arena Leaderboard. https://artificialanalysis.ai/text-to-image/arena?tab=leaderboard [Accessed 09-July-2025].

[9] Anthropic. 2024. Introducing the Model Context Protocol. https://www.anthropic.com/news/model-context-protocol [Accessed 09-July-2025].

[10] Shahar Avin, Haydn Belfield, Miles Brundage, Gretchen Krueger, Jasmine Wang, Adrian Weller, Markus Anderljung, Igor Krawczuk, David Krueger, Jonathan Lebensold, Tegan Maharaj, and Noa Zilberman. 2021. Filling gaps in trustworthy development of AI. *Science* 374, 6573 (2021), 1327–1329.

[11] Mike Belshe, Roberto Peon, and Martin Thomson. 2015. Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540. https://doi.org/10.17487/RFC7540

[12] David Belson. 2024. Cloudflare 2024 Year in Review. https://blog.cloudflare.com/radar-2024-year-in-review/ [Accessed 09-July-2025].

[13] Mike Bishop. 2022. HTTP/3. RFC 9114. https://doi.org/10.17487/RFC9114

[14] Miles Brundage, Shahar Avin, Jasmine Wang, Haydn Belfield, Gretchen Krueger, Gillian Hadfield, Heidy Khlaaf, Jingying Yang, Helen Toner, Ruth Fong, et al. 2020. Toward trustworthy AI development: mechanisms for supporting verifiable claims. *arXiv preprint arXiv:2004.07213* (2020).

[15] Cloudflare. 2025. Cloudflare Radar, Adoption & Usage. https://radar.cloudflare.com/adoption-and-usage [Accessed 09-July-2025].

[16] Ollama Contributors. 2025. Ollama: Local Large Language Model API. https://github.com/ollama/ollama/tree/main/docs Accessed: 2025-02-13.

[17] Francesco Croce, Christian Schlarmann, Naman Deep Singh, and Matthias Hein. 2025. Adversarially robust clip models can induce better (robust) perceptual metrics. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 636–660.

[18] Daniel Gomes de Pinho Zanco, Leszek Szczecinski, Eduardo Vinicius Kuhn, and Rui Seara. 2024. Stochastic analysis of the Elo rating algorithm in round-robin tournaments. *Digital Signal Processing* 145 (2024), 104313.

[19] Michael Dinzinger, Florian Heß, and Michael Granitzer. 2024. A Survey of Web Content Control for Generative AI. *arXiv preprint arXiv:2404.02309* (2024).

[20] Joel E Fischer. 2023. Generative AI considered harmful. In *Proceedings of the 5th international conference on conversational user interfaces*.

[21] Python Software Foundation. 2024. *asyncio - Asynchronous I/O framework*. https://docs.python.org/3/library/asyncio.html Version 3.12.

[22] Guo Freeman, Douglas Zytko, Afsaneh Razi, Cliff Lampe, Heloisa Candello, Timo Jakobi, and Konstantin Kosta Aal. 2025. New Opportunities, Risks, and Harm of Generative AI for Fostering Safe Online Communities. In *Companion Proceedings of the 2025 ACM International Conference on Supporting Group Work*. 2–5.

[23] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. 2023. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344* (2023).

[24] Yunhao Ge, Xiaohui Zeng, Jacob Samuel Huffman, Tsung-Yi Lin, Ming-Yu Liu, and Yin Cui. 2024. Visual Fact Checker: Enabling High-Fidelity Detailed Caption Generation. arXiv:2404.19752 [cs.CV] https://arxiv.org/abs/2404.19752

[25] Google Developers. [n. d.]. Manifest - Sandbox | Chrome Extensions. https://developer.chrome.com/docs/extensions/reference/manifest/sandbox. [Accessed 09-July-2025].

[26] Shayan Ali Hassan, Danish Humair, Ihsan Ayyub Qazi, and Zafar Ayyub Qazi. 2024. Rethinking Image Compression on the Web with Generative AI. *arXiv preprint arXiv:2407.04542* (2024).

[27] Wenji He, Haipeng Yao, Xiaoxu Ren, Yuanling Liu, Zehui Xiong, and Dusit Niyato. 2025. Advancing End-to-End Programmable Networks: Exploring the Interplay of Generative AI with Opportunities and Challenges. *IEEE Network* (2025).

[28] Ziyao He, Syed Fatiul Huq, and Sam Malek. 2025. Enhancing Web Accessibility: Automated Detection of Issues with Generative AI. *Proceedings of the ACM on Software Engineering* 2, FSE (2025), 2264–2287.

[29] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718* (2021).

[30] Yiqing Hua, Shuo Niu, Jie Cai, Lydia B Chilton, Hendrik Heuer, and Donghee Yvette Wohn. 2024. Generative AI in user-generated content. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–7.

[31] Fahime Khoramnejad and Ekram Hossain. 2025. Generative AI for the optimization of next-generation wireless networks: Basics, state-of-the-art, and open challenges. *IEEE Communications Surveys & Tutorials* (2025).

[32] Akio Kodaira, Chenfeng Xu, Toshiki Hazama, Takanori Yoshimoto, Kohei Ohno, Shogo Mitsuhori, Soichi Sugano, Hanying Cho, Zhijian Liu, Masayoshi Tomizuka, and Kurt Keutzer. 2025. StreamDiffusion: A Pipeline-level Solution for Real-time Interactive Generation. arXiv:2312.12491 [cs.CV] https://arxiv.org/abs/2312.12491

[33] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. 2025. FLUX.1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space. arXiv:2506.15742 [cs.GR] https://arxiv.org/abs/2506.15742

[34] Baolin Li, Rohan Basu Roy, Daniel Wang, Siddharth Samsi, Vijay Gadepally, and Devesh Tiwari. 2023. Toward Sustainable HPC: Carbon Footprint Estimation and Environmental Implications of HPC Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '23)*. Article 19, 15 pages.

[35] Siyuan Li, Xi Lin, Yaju Liu, Gaolei Li, and Jianhua Li. 2024. OpticGAI: Generative AI-aided Deep Reinforcement Learning for Optical Networks Optimization. In *Proceedings of the 1st SIGCOMM Workshop on Hot Topics in Optical Technologies and Applications in Networking*. 1–6.

[36] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. 2023. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Advances in Neural Information Processing Systems* 36 (2023), 20662–20678.

[37] Yinqiu Liu, Hongyang Du, Dusit Niyato, Jiawen Kang, Zehui Xiong, Yonggang Wen, and Dong In Kim. 2025. Generative ai in data center networking: Fundamentals, perspectives, and case study. *IEEE Network* (2025).

[38] Jialun Lyu, Jaylen Wang, Kali Frost, Chaojie Zhang, Celine Irvene, Esha Choukse, Rodrigo Fonseca, Ricardo Bianchini, Fiodar Kazhamiaka, and Daniel S. Berger. 2023. Myths and Misconceptions Around Reducing Carbon Embedded in Cloud Platforms. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems (HotCarbon '23)*. Article 7, 7 pages.

[39] Shweta Mahajan, Tanzila Rahman, Kwang Moo Yi, and Leonid Sigal. 2024. Prompting hard or hardly prompting: Prompt inversion for text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6808–6817.

[40] Filippo Menczer. 2002. Growing and navigating the small world web by local content. *Proceedings of the National Academy of Sciences* 99, 22 (2002), 14014–14019.

[41] Netflix. [n. d.]. How to control how much data Netflix uses. https://help.netflix.com/en/node/87 [Accessed 14-October-2025].

[42] NVIDIA. 2024. RTX Video SDK. https://developer.nvidia.com/rtx-video-sdk [Accessed 14-October-2025].

[43] Atila Orhon, Aseem Wadhwa, Youchang Kim, Francesco Rossi, Vignesh Jagadeesh, et al. 2022. Deploying transformers on the apple neural engine. *Apple Machine Learning Reseach, Computer Vision, research areaSpeech and Natural Language Processing, Highlight June* (2022).

[44] Sushil K Oswal and Hitender K Oswal. 2024. Examining the accessibility of generative AI website builder tools for blind and low vision users: 21 best practices for designers and developers. In *2024 IEEE International Professional Communication Conference (ProComm)*. IEEE, 121–128.

[45] Satadru Pan, Theano Stavrinos, Yunqiao Zhang, Atul Sikaria, Pavel Zakharov, Abhinav Sharma, Mike Shuey, Richard Wareing, Monika Gangapuram, Guanglei Cao, et al. 2021. Facebook's tectonic filesystem: Efficiency from exascale. In *19th USENIX Conference on File and Storage Technologies (FAST 21)*. 217–231.

[46] The Chromium Projects. 2025. *Chromium - The Open-Source Project Behind Google Chrome.* https://www.chromium.org/Home/ [Accessed 09-July-2025].

[47] Qualcomm. 2024. Unlocking on-device generative AI with an NPU and heterogeneous computing. https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/Unlocking-on-device-generative-AI-with-an-NPU-and-heterogeneous-computing.pdf/ [Accessed 14-October-2025].

[48] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[49] Samsung. 2025. Samsung's Pivotal Role in Pioneering On-Device Generative AI, Tech Report. https://semiconductor.samsung.com/news-events/tech-blog/samsungs-pivotal-role-in-pioneering-on-device-generative-ai/ [Accessed 14-October-2025].

[50] Swamit Tannu and Prashant J. Nair. 2023. The Dirty Secret of SSDs: Embodied Carbon. *SIGENERGY Energy Informatics Review* 3, 3 (Oct. 2023), 4–9.

[51] Telefonica. 2025. Smartphones in 2024: how many there are, usage time or main uses. https://www.telefonica.com/en/communication-room/blog/smartphones-2024/ [Accessed 21-October-2025].

[52] Martin Thomson and Cory Benfield. 2022. HTTP/2. RFC 9113. https://doi.org/10.17487/RFC9113

[53] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. 2022. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers.

[54] Risqo Wahid, Joel Mero, and Paavo Ritala. 2023. Written by ChatGPT, illustrated by Midjourney: generative AI for content marketing. *Asia Pacific Journal of Marketing and Logistics* 35, 8 (2023), 1813–1822.

[55] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, and Tat-Seng Chua. 2025. Generative Recommendation: Towards Personalized Multimodal Content Generation. In *Companion Proceedings of the ACM on Web Conference 2025*. 2421–2425.

[56] Jakub Wąsala, Bartłomiej Wrzalski, Kornelia Noculak, Yuliia Tarasenko, Oliwer Krupa, Jan Kocoń, and Grzegorz Chodak. 2025. Enhancing AI Face Realism: Cost-Efficient Quality Improvement in Distilled Diffusion Models with a Fully Synthetic Dataset. In *International Conference on Computational Science*. Springer, 119–134.

[57] Thiong'o Waweru and tridens technology. 2025. *Mobile Data Statistics 2025: Global Usage Trends and Consumption.* https://tridenstechnology.com/mobile-data-statistics/ [Accessed 21-October-2025].

[58] Rongyuan Wu, Lingchen Sun, Zhiyuan Ma, and Lei Zhang. 2024. One-step effective diffusion network for real-world image super-resolution. *Advances in Neural Information Processing Systems* 37 (2024), 92529–92553.

[59] Minrui Xu, Hongyang Du, Dusit Niyato, Jiawen Kang, Zehui Xiong, Shiwen Mao, Zhu Han, Abbas Jamalipour, Dong In Kim, Xuemin Shen, et al. 2024. Unleashing the power of edge-cloud generative AI in mobile networks: A survey of AIGC services. *IEEE Communications Surveys & Tutorials* 26, 2 (2024), 1127–1170.

[60] Eric York. 2023. Evaluating chatgpt: Generative ai in ux design and web development pedagogy. In *Proceedings of the 41st ACM international conference on design of communication*. 197–201.

[61] Yang Zhao, Yanwu Xu, Zhisheng Xiao, Haolin Jia, and Tingbo Hou. 2024. Mobilediffusion: Instant text-to-image generation on mobile devices. In *European Conference on Computer Vision*. Springer, 225–242.